

The Gherkin Language

Jason Roberts
@robertsjason
DontCodeTired.com



pluralsight 
hardcore developer training

Module Overview

- **Features and scenarios**
- **Given, When, Then steps**
- **Tags**
- **Comments**
- **Using data tables in steps**
- **Scenario Outlines**
- **Backgrounds**

What is Gherkin?

- Business readable domain specific language
- Represents tests in natural language, not code
- Line-oriented
- Uses indentation to create structure
- Keywords (e.g. Feature, Scenario)
- Localised in 40+ spoken languages (French, Bulgarian, Japanese, etc.)

Features

- **Contain one or more scenarios**
- **Group/contain logically related test scenarios**
- **Represent small, discrete features of the system**

Features

- Start with “Feature: ” keyword
- Followed by feature name/terse description
- Optional free text description

Feature: Brush Teeth

*Brushing teeth is good
because it helps keep
them clean*

Features

- As a... I want... So that...

Feature: Brush Teeth

As a human

I want to brush my teeth

So that they stay pain free



Role / person
centric

Features

- In order to... As a... I want...

Feature: Brush Teeth

In order to stay pain free

As a human

I want to brush my teeth



Value
centric

Scenarios

- Concrete examples of expected system behaviour
- Each scenario describes a particular situation
- Each scenario should be independent and isolated
- Can represent:
 - “Happy paths”
 - Error paths
 - Edge cases
- Start with: “Scenario: ” keyword
- Followed by title

Scenario: Successful brushing

Scenario Steps

*Set up
initial state*

Given

*Perform
action(s)*

When

*Check
end state*

Then

Scenario Steps

Scenario: Successful brushing

Given there is toothpaste on the brush

Given the mouth is open

When the back teeth are brushed

When the front teeth are brushed

Then the teeth look clean

Then the mouth feels fresh

Then the braces aren't damaged

Scenario Steps

Scenario: Successful brushing

Given there is toothpaste on the brush

And the mouth is open

When the back teeth are brushed

And the front teeth are brushed

Then the teeth look clean

And the mouth feels fresh

But the braces aren't damaged

Scenario Steps

Scenario: Successful brushing

Given there is toothpaste on the brush

And the mouth is open

When the back teeth are brushed

And the front teeth are brushed

Then the teeth look clean

And the mouth feels fresh

But the braces aren't damaged

Tags

- Mark features and scenarios with arbitrary tags
- Map to unit test framework “categories”
- Scenarios “inherit” feature tags
- Can have multiple tags
- Tags specified using @
- @ignore is a special case

@smoketest

Scenario: Successful brushing

...

Tags

- **Possible uses:**

- Group features into feature supersets
- Mark certain tests as @important
- Differentiate between @slow and @fast executing tests
- Mark tests as @wip
- Mark tests to be executed @weekly @daily @hourly
- Mark tests as @humanexecuted

Using Data Tables in Steps

- Allows tabular data to be passed to an automation step
- Useful for specifying larger data rather than multiple steps

Scenario: Successful brushing

Given I have paste

And brush

And water

When the back teeth are brushed

Then the teeth look clean

Using Data Tables in Steps

Scenario: Successful brushing

Given I have the following tools:

toolname
paste
brush
water

When the back teeth are brushed

Then the teeth look clean

Scenario Outlines

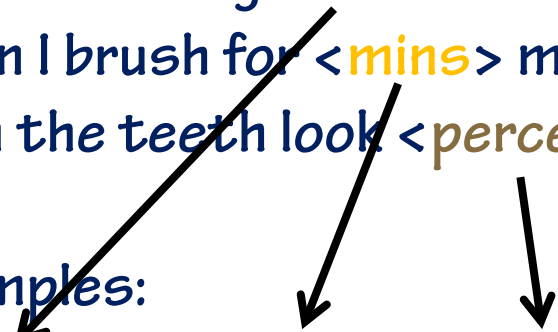
Scenario Outline: Teeth whiteness

Given I'm using "<brand>" brand toothpaste

When I brush for <mins> minutes

Then the teeth look <percent> white

Examples:



brand	mins	percent	
Brand X	1	80	
Brand Y	3	100	
Brand Z	10	50	

Background

- Provides context (state setup) to the scenarios in a feature
- Executed before every scenario

Background:

Given I have teeth

And I have some toothpaste

Scenario: Successful brushing

...

Scenario: Toothpaste runs out

...

Module Summary

- **Features and scenarios**
- **Given, When, Then steps**
- **Tags**
- **Comments**
- **Using data tables in steps**
- **Scenario Outlines**
- **Backgrounds**